Design and Implementation of a DVB-H IP Decapsulator

Wen-Tzeng Huang, Sun-Yen Tan, Chin-Hsing Chen, and Ren-Guey Lee

***Abstract*—**The digital video broadcasting-handheld (DVB-H) standard determined by the European Telecommunications Standard Institute is based on the terrestrial broadcast system DVB-Terrestrial (DVB-T), which has allowed digital video to be incorporated into handheld device applications. DVB-H also makes coded orthogonal frequency division multiplexing, time slicing, and multi-protocol encapsulation-forward error correction (MPE-FEC) compatible with the existing broadcasting standard. We describe a DVB-H Internet Protocol (IP) decapsulator that can achieve time slicing and MPE, as well as cross platform compatibility, and is highly adaptable. Power consumption is always an issue in handheld devices, and time slicing effectively reduces the most serious DVB-H demodulator power consumption problems. The DVB-H protocol uses MPE to deliver information, and unlike other DVB protocols, the video and sound data of DVB-H are not packed directly into the MPEG-2 transport stream. Instead, they are packed into the IP datagram, MPE is performed, and then the data are delivered via the MPEG-2 transport stream. In this way, DVB-H is backward-compatible with DVB-T. We implemented the DVB-H IP decapsulator using pure Java language, which enables it to function across platforms. We specified a platform-adaptive layer to reduce the drive complexity between the Java virtual machine and the demodulator. Because DVB-H has a built-in IP decapsulator transport stream dispatcher module, it independently creates program-specific information and service information subdecoders. Therefore, the packet that recognizes the decoders can be obtained from the transport stream dispatcher while the registration of that specific packet is completed.

***Keywords*—**DVB-H, Time-slicing, Multi-Protocol Encapsulation, IP-Decapsulator.

I. INTRODUCTION

THE digital video broadcasting-handheld (DVB-H) system standard [1] is based on a terrestrial broadcast system called DVB-terrestrial (DVB-T) [2], but applying DVB-T systems to handheld receivers may lead to problems with mobile receiving ability and power consumption. DVB-H has additional features, such as transmission parameter signaling (TPS), 4K mode and in-depth symbol interleaves, a multi-protocol encapsulation-forward error correction (MPE-FEC) encoding mechanism, time-slicing techniques, and a soft-handover mechanism, which improve mobile reception and reduce power consumption. These features allow handheld terminals to receive television signals.

A DVB-H standard is still being established, and most DVB research has focused on the design of a whole system [3][8] and on performance evaluation [4][9]. Fig. 1 shows a conceptual link-layer block diagram [3] of a DVB-H system. The main tasks of the link layer are Internet Protocol (IP) decapsulation and error correction. At the receiving end within the link layer, a demultiplexer (demux) analyzes the transport stream, and then sends IP datagrams to the back end, allowing application programs to broadcast audio and video streams. To reduce power consumption, the streams at the receiving end also contain the power control information of the physical layer, which can be analyzed via the link layer and sent to the power control module. Gerard evaluated the implementation and power consumption performance of a DVB-H system [4], but because DVB-H is a new research area, few hardware implementations are available. This paper presents a software design method for an IP decapsulator.

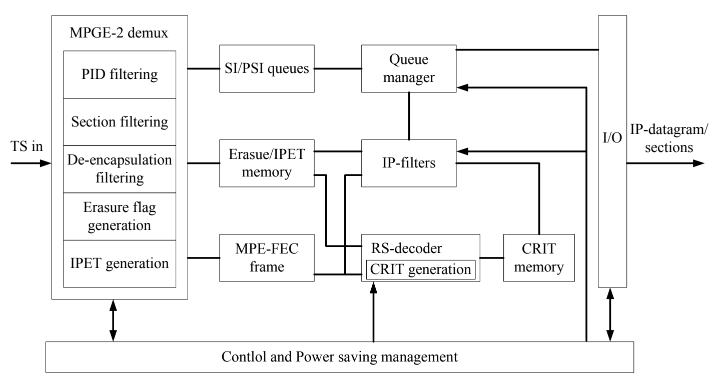

Fig. 1 Conceptual link-layer block diagram

The goal of our study was to design and implement a DVB-H IP decapsulator using pure Java language to enable functionality across platforms. We used a platform-adaptive layer to reduce the driving complexity between a Java virtual machine and the demodulator. Because DVB-H has a built-in IP decapsulator transport stream dispatcher module, it

Manuscript received June 30, 2007; Revised September 23, 2007.
Wen-Tzeng Huang is with the Electronic Engineering Department, National Taipei University of Technology, Taipei, Taiwan (e-mail: wthuang@ntut.edu.tw).
Sun-Yen Tan is with the Electronic Engineering Department, National Taipei University of Technology, Taipei, Taiwan (phone: 886-2-27712171; fax: 886-2-27317120; e-mail: sytan@ntut.edu.tw).
Chin-Hsing Chen is with the Department of Management Information Systems, Central Taiwan University of Science and Technology, Taichung, Taiwan (e-mail: chchen@ctust.edu.tw).
Ren-Guey Lee is with the Electronic Engineering Department, National Taipei University of Technology, Taipei, Taiwan (e-mail: evans@ntut.edu.tw).

independently creates program-specific information and service information subdecoders. Therefore, the packet that recognizes the decoders can be obtained from the transport stream dispatcher while the registration of that specific packet is completed. The above mechanism allows the new decoding module to be expanded with existing resources when more information tables are added in the future.

A television broadcast may contain video, audio, and text tracks. Multimedia streams are also multiplexed streams. As the data are received, the multiplexed streams must be demultiplexed into video and audio. For transmission, DVB-H encapsulates IP datagrams into transport streams, and then demultiplexes the streams into IP datagrams that can be received. Fig. 2 shows an IP service transport system structure. A DVB-H IP encapsulator is used to pack the DVB-H service into an IP datagram, followed by MPE and time slicing. Then, a DVB-H IP decapsulator is used to recover information at the receiving end.

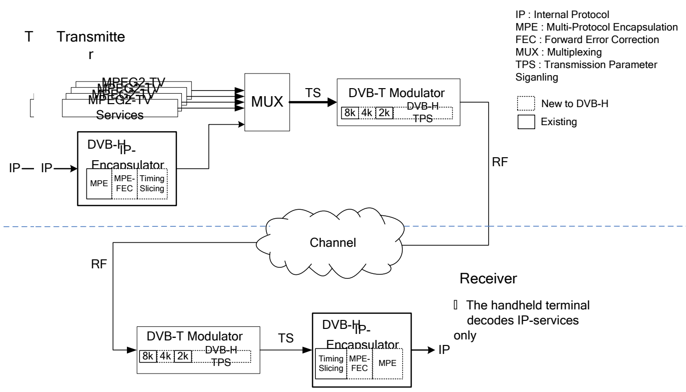

Fig. 2 DVB-H IP transport system structure

Time slicing reduces both the average energy consumption of handheld terminals and communications between base stations. Fig. 3 shows a burst transmission.

As shown in Fig. 4, a time tag (Delta-T) within the current burst indicates the arrival time of the next receiving burst. No transmission takes place between the two bursts. By sharing bandwidth, the burst is able to deliver the streams of various services. Time-slicing techniques also allow the receiving end to monitor the signals of nearby base stations, leading to a soft handover. Jitter in the Delta-T indicates the time necessary to turn on the demodulator in advance at the receiving end. The maximum burst duration is the maximum burst in terms of time. Using these three time parameters, the IP datagram streams can be received correctly via time slicing.

Section 2 describes the DVB-H IP decapsulator and its implementation. Section 3 presents the experimental results. Section 4 presents the hardware testing environment and the test results, and Section 5 provides our conclusions.

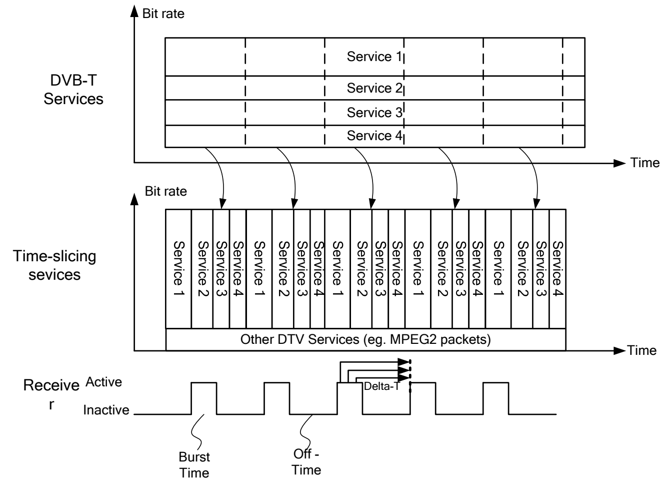

Fig. 3 Burst transmission

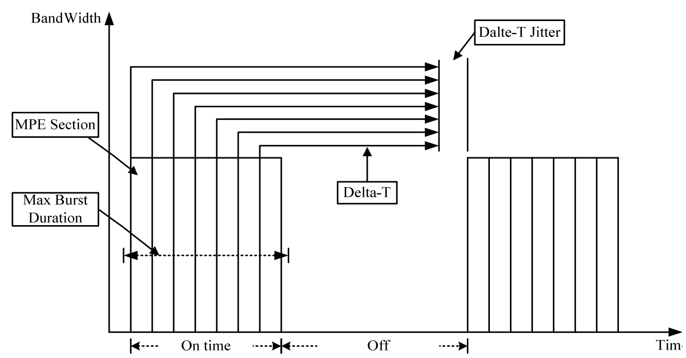

Fig. 4 Burst time parameters

II. DVB-H IP DECAPSULATOR

Fig. 5 shows the software structure of a DVB-H IP decapsulator. The decapsulator consists of the input interface of the platform adaptive layer, the output interface of the platform adaptive layer, the transport stream dispatcher, subdecoders, and the controller.

The DVB IP decapsulator is not directly connected to the front demodulator interface, and is only needed to implement the platform adaptive layer. This allows communication between various stream source interfaces.

A platform adaptive layer at the video output end connects the DVB-H IP decapsulator and available player programs. The output-end interface is easily achieved using network sockets, files, and pipelines in the platform adaptive layer, increasing the output options.

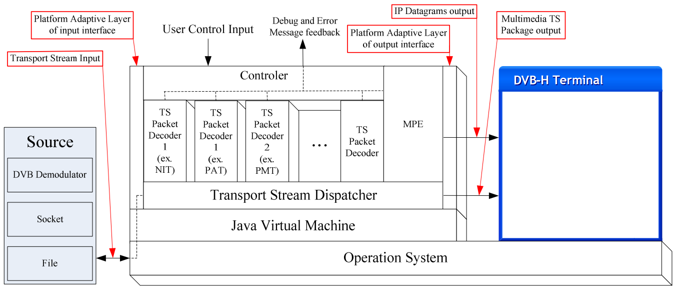

Fig. 5 Soft structure of a DVB-H IP decapsulator

TABLE I
ALGORITHM FOR THE TRANSPORT STREAM DISPATCHER

```
If SystemStart then
{
   Set Packet = ReadPacketFromInput(Buffer);
   if PID equal 8191 then
        continue;
   if PayloadUnitStartIndicator equal 1 then {
        if ContinueSection not equal NULL then
              call ContinuityDecodeFunction(Packet);
              continue;
        Set Decoder = FindDecoder (TID<<16&PID);
        if Decoder equal NULL then
              call UnknownPacket ();
        else
              call DecodeFunction(Packet);
   }
   else
   {
        if ContinuityPacket not equal NULL then
              call ContinuityDecodeFunction (Packet);
        else
              OutputToDVB-Tchannel (Packet);
   }
}
```

A. Transport stream dispatcher

Various streams of information, including video, the program specific information (PSI)/service information (SI) control table, and the MPE packet, are delivered within the DVB-H transport stream. The transport stream dispatcher then delivers the packets to various process modules depending on their type. It can also filter programs specified by the user from the packets before passing them to the DVB-H terminal device. The dispatcher reads the packets from the source end and then identifies the header of the transport packet. The preprocessing operations shown in Table 1 will be executed based on the header information.

During burst transport, the stream may contain multiple filled packets. By identifying these filled packets during reception, the transport stream dispatcher can reduce processing time. That is, the filled packets are discarded rather than being sent to the subdecoder to consume processing time. To simplify the subdecoder, the transport stream dispatcher provides a dispatching table to register the decoded packet types. The dispatching table is a type of hashing table application in which time is a constant regardless of the number of registrations. Therefore, the dispatching table can support new tables or the private tables of television stations.

TABLE II
ALGORITHM OF THE SUBDECODER

```
Function: DecodeFunction(Packet)
// get the first packet inside the section from transport
// stream dispatcher and decode it.
Set Payload = PayloadOfThePacket;
Set PayloadLength = PacketPayloadLength;
if sectionLength equal PayloadLength then {
      DecodeHeader();
      if Payload not equal NULL then
        OutputToDVB-HChannel (Payload);
}
else
{
      Set ReceiveLength equal PayloadLength;
      Set ContinuityPacketPointer(This);
}


Function: ContinueFunction(Packet)
// get the subsequent sections from the transport stream
// dispatcher.
Set Payload = Payload + PacketPayload;
Set PayloadLength = PayloadLength + RcvLength;
if ReceiveLength equal section_length then
{
      DecodeHeader();
      Set ContinuityPacketPointer = NULL;
}
```

B. Subdecoder elements

During initialization, the subdecoder registers the packet type with the transport stream dispatcher to obtain the corresponding packets. The transport stream dispatcher reduces the items implemented by the subdecoder, which may include initialization, decoding, and packet loss. A common subdecoder algorithm is shown in Table 2. Depending on the registered information, the dispatcher delivers the packets that are registered to each corresponding packet type. Based on the information in the packet, the subdecoder releases configurations or messages to the controller objects. Delivered messages contain notices that include process records, debugging messages, and error message reports.

Based on the length field of the session, the subdecoder can determine when a message exceeds the carrying capacity of a packet; in this case, it sets the continuity packet pointer object to itself. After receiving a packet, the transport stream dispatcher checks whether the continuity packet pointer object is null, and if so, the subdecoder handling the received packet is found in the dispatching table. Otherwise, the packet is delivered to the continuity receiving subdecoder, and the operations are repeated until the receiving session is complete. Then the subdecoder resets the continuity packet pointer object to null, and the procedure is followed to find the corresponding subdecoder of the packet.

C. *The controller objects*

The control information entered by the user is received using the controller interface. This differs from the conventional method in which the control interface is embedded in the player, and separates the DVB-H IP decapsulator and the player device. The main functions of the controller objects involve decapsulator timing control, the user control input interface, and the message output interface. Depending on the control requests entered by the user, the controller releases control messages to various control modules; the requests include playing, tuning, and termination. Each module is adjusted by the released messages to meet the requested operations of the user, and the control object is an interactive interface between the DVB-H IP decapsulator and the user. In a handheld device, a graph interface can be used to show the information, and the keyboard to receive messages from the user. In addition to catching user input, the controller serves as the message output interface. The controller is only interactive with the subdecoder, although it is connected to all of the elements within the DVB-H IP decapsulator to receive output messages.

D. *Time slice mechanism*

During the processing of the data-linking layer, it is necessary to transfer the Delta-T and maximum burst duration parameters to the physical layer in real time.

As shown in Fig. 6, setting buffers can reduce the real-time tasks at the receiver end. However, the Delta-T parameter must be handled in real time. Delta-T information is contained within each MPE section, but when the MPE section is buffered, the Delta-T information may expire if the time required to fetch the MPE section from the buffers exceeds the time allowed by the Delta-T jitter. Therefore, the Delta-T information becomes useless.

The first MPE section has the shortest delay. When all of the MPE sections within the previous burst are processed completely, the sections within the next burst are transferred into the buffers via the physical layer and are directly accessed by the DVB-H IP decapsulator. The Delta-T information that must be fetched from the first MPE section is within the burst. The arrival time of the next burst to the physical layer is necessary for the start time of the burst to be set; the Delta-T information within the remaining MPE section does not need to be read.

To finish decoding the DVB-H information, all of the MPE sections within the buffer must be processed before the next burst arrives. Then the first MPE section can be received in real time at its starting point.

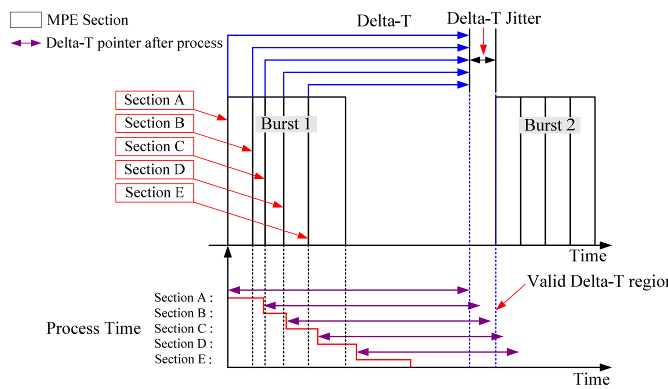

Fig. 6 Handling the delay time for a packet

III. Experimental Results

We used a WAFER-9371A industrial computer manufactured by IEI Technology Corporation as the basic working platform to simulate a personal digital assistant (PDA) handheld application device. The transport stream supported by the broadcasting project of public television and the original IP stream were used to verify accuracy. Table 3 shows the information captured from the public television image source. We achieved a 79.7% savings in power consumption. A different number of MPE sections was delivered inside each burst, that is, between 99 and 486 packets (for the test case only).

TABLE III
Public Television Video Parameters

Parameters	Values
Packet number	4,019,179 (pieces) (188 bytes/each packet)
Size of file	755,605,652 (bytes)
Burst number	552 (times)
MPE section number	21569 (pieces)
Delta-T	1250 ms
Delta-T jitter	7.5 ms
Max. burst continuous time	200 ms
Power saving rate	79.7%
Video resolution	176*120 (QCIF)
Max. average bit rate	512 kbps
Frame refresh rate	15 fps

The main tasks of the transport stream dispatcher module are decoding the transport stream packet, filtering filled packets, and finding the corresponding subdecoders for the packets using the hashing table operation. The hashing table operation is executed quickly, with an average executing time of 884 ns.

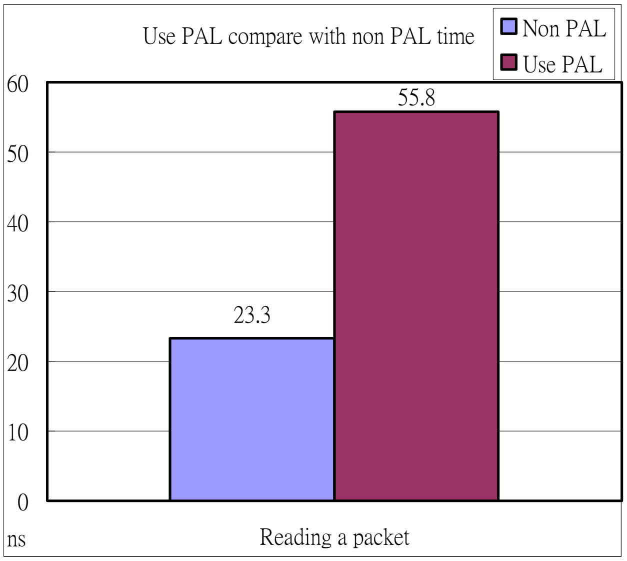

Fig. 7. Packet reading time with and without the platform adaptive layer

Fig. 7 indicates that the time difference between reading a packet with the platform adaptive layer and obtaining a packet directly from the virtual machine was only 32.5 ns, which is negligible compared to the total processing time. The hardware input/output time is usually greater.

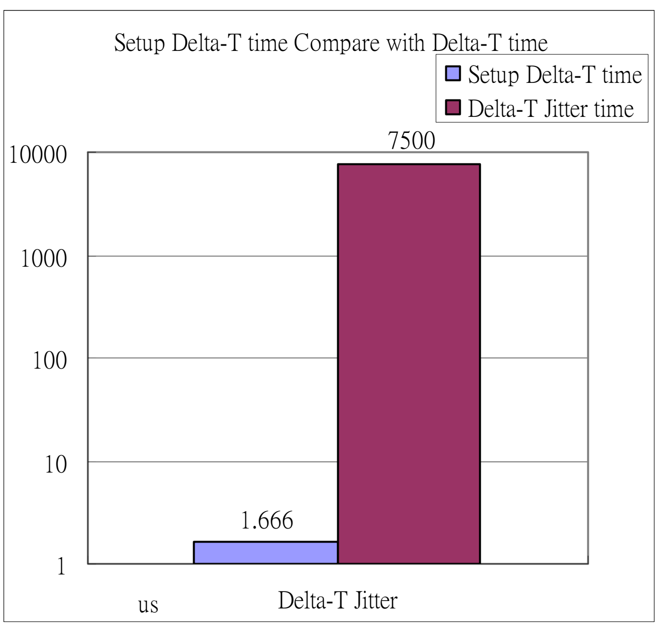

Fig. 8. Delta-T executing time setup

Fig. 8 shows the required time for establishing Delta-T. The hashing table operation of the transport stream dispatcher required the most time. Therefore, the total time was slightly greater than the transport stream dispatcher hashing table operation time.

Fig. 9 presents the burst test results, including the decoding times for all of the 552 bursts in the file. As shown in the figure, in approximately 83% of bursts, the processing times were between 50 and 70 ms. All of the bursts, including operations that used the platform adaptive layer and those that did not, finished the processes within 120 ms.

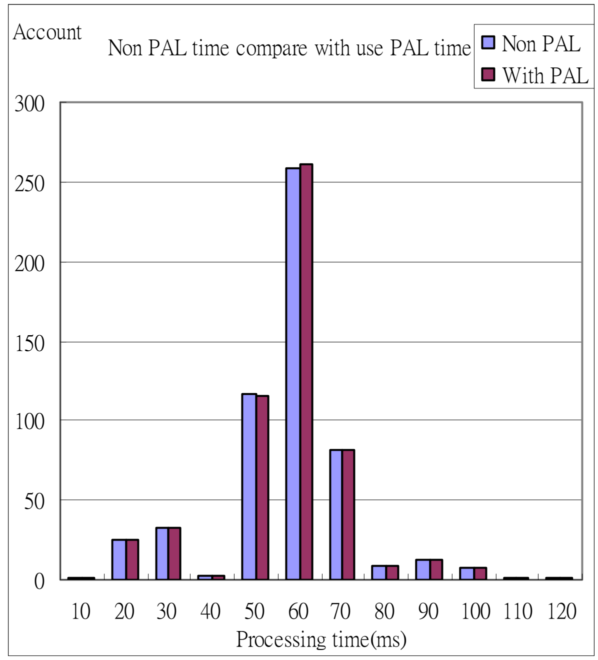

Fig. 9 Burst processing time

Some differences occurred in the test results that finished between 50 and 60 ms, but the delivery time was generally insignificant compared to the hardware input/output time. Therefore, the total difference in performance is very small when the platform adaptive layer is used.

Due to the optimization of the compilers and uncertainties in the operating systems scheduler, the processing times of some bursts were smaller than 10 ms, while other bursts took up to 120 ms to process. This meets the Delta-T specifications (1250 ms) for delivery from public television.

The maximum average bit rate was only 512 kbps. The software uses bytes as processing data units, so it is only necessary to process 64 kb of data per second. The processors of current handheld devices have rates between 300 and 600 MHz, so processing the data should not become an overhead issue of the receiver. The image shown in Fig. 10 was decoded by the DVB-H IP decapsulator described in this paper. The program image was broadcasted over public television, and had a resolution of 176*120. It was played at its original size, matching the screen size of current handheld devices.

Fig. 10 Played program image

TABLE IV
LIST OF HARDWARE COMPONENTS

Name of hardware components	Manufacturer and model
Modulator	DVB-T/H
ASI conversion card (DVB ASI output)	DTA-100
RF antenna module	SD10
DVB-H receiver antenna	DiBcom DVB-H receiver
PDA	***ASUS COMPUTER*** MyPal A636

TABLE V
ASUS COMPUTER MYPAL A636 SYSTEM SPECIFICATION

CPU processor	XScale 416 MHz
Operating system	Microsoft Windows Mobile 5.0
Memory	128 MB Flash ROM/64 MB SDRAM
Expansion slots	SDIO Slots

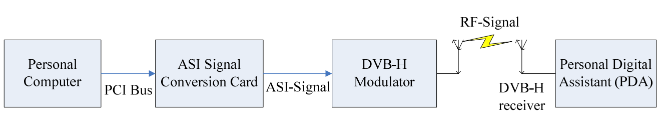

Fig. 11 Flowchart of the broadcasting experiment

IV. THE HARDWARE TESTING ENVIRONMENT

A personal computer was used at the transmission end to open the transport stream file recorded from public television. An ASI conversion card was used to convert the DVH-H transport stream file into ASI signals that were sent to the DVB-H modulator. The ASI signals were modulated into radio frequency (RF) signals via the DVB-H modulator, and the modulated RF signals were transmitted using the RF antenna.

At the receiver end, a DVB-H receiver antenna received the signals. Our software-decoding mechanism decoded the transport streams, and the decoded images were played on the PDA. Table 4 provides a list of the hardware components used in this experiment. Table 5 lists the specifications of the hardware components. Fig. 11 presents a flowchart of the broadcasting experiment. A picture of the hardware components for the experiment is shown in Fig. 12, and the real-time transmitted and decoded images are given in Fig. 13. The DVB-H modulator converted ASI signals into RF signals, and then the RF signals were transmitted and broadcasted to the PDA via the RF antenna. The PDA received the transmitted signals using a DVB-H receiver antenna, decoded the received signals, and played the decoded images.

Fig. 12 Experimental hardware components and DVB-T/H modulator

Fig. 13 Real-time transmitted and decoded images

V. CONCLUSION

In this paper, we proposed a DVB-H IP decapsulator mechanism that processes data streams delivered via public television. Minimal differences were observed between the DVB-H IP decapsulator processing times for whole bursts with and without the platform adaptive layer. At a maximum, both DVB-H IP decapsulators finished processing all of the MPE sections within 120 ms. The processing time accounted for only 9.6% of the Delta-T, and therefore met the requirement of decoding all MPE sections within the Delta-T.

A real hardware testing environment was established to test the proposed decapsulator, and the decoded images were played successfully on a PDA. Fetching Delta-T from the first MPE section within the burst took 1.666 μs, or 0.02% of the time requirement of 7.5 ms, leaving enough time for the physical layer to prepare to turn on the demodulator before the arrival of the next burst.

The DVB-H IP decapsulator is an important component at the receiver end, and used to link services between the upper layers and the physical layer. When implemented, this data-linking layer can be used as a research platform to develop multiple-service applications in the future. We anticipate the design and implementation of the data-linking layer in this research to produce additional DVB-H applications and designs, make handheld televisions possible, and bring abundant information resources to the users of handheld devices.

References

[1] DVB Document A092, "DVB-H Implementation Guidelines," *European Telecommunications Standards Institute*.

[2] TS 101 154, "Digital Video Broadcasting (DVB); Implementation Guidelines for the Use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream," *European Telecommunications Standards Institute*.

[3] O. Eerenberg, A. Koppelaar, A. M. Stuivenwold, P. H. N. de With, "IP-Recovery in the DVB-H Link Layer for TV on Mobile," *IEEE International Conference on Consumer Electronics*, 2006.

[4] F. Gerard, "DVB-H: Digital TV in the Hands!," *International Exhibition and Conference: Conference and Business Seminars Program*, 2005.

[5] ETSI EN 302 304, "Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H)," *European Telecommunications Standards Institute*.

[6] EN 101 162, "Digital Video Broadcasting (DVB); Allocation of Service Information (SI) and Data Broadcasting Codes for Digital Video Broadcasting (DVB) Systems," *European Telecommunications Standards Institute*.

[7] EN 300 468, "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB system," *European Telecommunications Standards Institute.*

[8] S. Drude, M. Klecha, "System Aspects for Broadcast TV Reception in Mobile Phones," *IEEE International Conference on Consumer Electronics*, 2006.

[9] E. D. Balaguer, F. H. P. Fitzek, O. Olsen, "Performance Evaluation of Power Saving Strategies for DVB-H Services Using Adaptive MPE-FEC Decoding," *The 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications*, 2006.